LEVEL 3 FARM INSTALLATION

Nuno Leonardo

Massachusetts Institute of Technology

for the EVB+LEVEL3+ONLINE teams

October 2003

Level 3 Farm Installation

N. Leonardo

Massachusetts Institute of Technology

Abstract

This document describes a kickstart installation procedure put together for the Level3 online farm at CDF. Both floppy and hard-drive based installation procedures have been implemented and are here described in detail.

Contents

1 Installation

A full node system installation becomes necessary whenever one has a new or blank hard drive, or an Operating System upgrade is to be performed.

Installation of a farm node can be achieved using an adapted recovery procedure. Indeed, such an operation has been repeatedly performed in Level3 in the past. Such a recovery floppy based installation may even be expedite in terms of time duration of the procedure, on a single node basis.

The task nevertheless of installing a large farm system like Level3 can be a challenging one. When the time comes for a full farm upgrade, for example, alternative ways may be worth examining. The so-called RedHat *kickstart* mechanism is explored and customized to farm requirements. A related floppy-*less* procedure for full farm installation is implemented.

1.1 Kickstart

Using individual Fermi RedHat Linux installation on multiple Level3 farm boxes repeatedly, as would be the case during OS upgrade, is time consuming and prone to errors and disparities.

A mechanism for automating the installs of a large number of nodes, which additionally have similar configurations, is therefore desirable to say the least. The *kickstart* [5] RedHat installation method provides in principle such a mechanism. Effectivelly, this allows the *scripting* of the otherwise interactive installation process; the latter is driven by a configuration file containing the answers to the questions that would normally be asked during a manual installation.

Most features can be specified in this single file, including: network configuration, distribution source and method, root password selection, boot loader, language, timezone, packages to be installed.

Additionally, the method offers the possibility of specifying a list of shell level commands (i.e. scripts) to be executed just before (`pre`) or after (`post`) the normal system installation. Use is made of these useful features for specific system and Level3 software installation.

As a general wisdom remark, it should be acknowledged the likelihood of sintax change and availability of even broader functionality in coming RedHat releases.

1.1.1 Image

The installation images can be dowloaded from appropriate locations [4]. The image contents can be accessed for customization purposes by transfering the kickstart image file `bootnet.img` to a device such as a floppy disk as follows.

```
fdformat  /dev/fd0
dd if=bootnet.img of=/dev/fd0H1440
mount -t msdos /dev/fd0 /mnt/floppy
```

```
ls /mnt/floppy/
  boot.msg
  general.msg
  initrd.img
  ldlinux.sys
  param.msg
  rescue.msg
  snake.msg
  syslinux.cfg
  vmlinuz
```

`vmlinuz` is the Linux kernel. The other larger file (`initrd.img`) is the initial root disk image (an ext2 filesystem compressed in a file, containing e.g. the collection of loadable kernel modules).

The file `syslinux.cfg` is the configuration file for the the *syslinux* boot loader, and the various `*.msg` files are message files which are normally displayed by the loader. The config file is relevant only for floppy based installation, in which case it needs to be customized; while the message files can be safely removed if floppy space is needed.

The process is guided by a kickstart configuration file (`ks.cfg`) which needs to be added together with the files mentioned above.

1.1.2 Configuring system installation

The appropriate kickstart configuration, **ks.cfg**, file needs to be created. This is a text file, containing a list of directives and keywords, and can be written from scratch. Alternatively, one can use the kickstart configurator application, `/usr/sbin/ksconfig`.

In the current Fermi Linux release a config file is automatically generated by anaconda; this file contains the configuration selected in the performed installation (done e.g. using a standard interactive installation), and is located at `/root/anaconda-ks.cfg` on the installed node.

The Kickstart config file is formed of the following main sections:

1. System information
2. RedHat packages to be installed
3. `pre` and `post` installation shell commands

Keywords must be in order; comment lines start with `#`.

Excerpts of the Level3 configuration file are persented next with explanations.

Networking

The installation is done via the network. A static network configuration method is used.

A set of `IP addresses` are *reserved* in the Level3 Ethernet network for use in the installation process. These are explicitly mentioned in the `/etc/hosts` file of the server; e.g.

```
192.168.23.250        b0l3ks0.fnal.gov  b0l3ks0
...
```

All the required networking information needs to be specified following the **network** keyword, in a single line.

```
network --bootproto static  --device eth0
        --ip 192.168.23.250 --hostname b0l3ks0
        --netmask 255.255.0.0 --nameserver 192.168.24.11
```

The specified information is static, it will be used during the installation process, as well as after the installation if not changed in the post configuration section.

The server from which to install and installation tree directory are specified with the **nfs** keyword.

```
nfs --server 192.168.24.11 --dir /scratch/731a.install/i386
```

Partitioning

First, disk partition table is initialized. The **zerombr** keyword clears the master boot record, removing exisiting OS boot loader. **clearpart** removes exisiting partitions, and initializes the disk label to the default.

```
zerombr yes
clearpart --all --initlabel
```

The new partitioning is specified via the **part** keyword. This allows to specify, for each partition, the mount point, filesystem type, minimum size, disk where it is to be created, among others. The following table is being used.

```
part /boot --fstype ext3 --size=50 --ondisk=hda --asprimary
part / --fstype ext3 --size=4096 --ondisk=hda --asprimary
part /tmp --fstype ext3 --size=2048 --ondisk=hda
part swap --size=2047 --ondisk=hda --asprimary
part /cdf --fstype ext3 --size=1 --grow --ondisk=hda
```

Other options

install; make a fresh system installation (rather than upgrade)

text; perform installation in text mode (default is graphical mode)

lang en_US; set language for installation (English)

langsupport en_US; set languages to install on the system

timezone –utc America/Chicago; set system timezone (see `timeconfig`)

bootloader; set boatloader and its location; defaults are GRUB (for LILO, `--useLilo`) at *mbr*

keyboard us; set system keyboard type

mouse none; do not configure mouse for the installed system

skipx; do not configure X on the installed system

authconfig; set up the authentication options for the system (see `authconfig`)

rootpw –iscrypted XA8wIytED41RI; set the system's root password, based on previosuly derived encrypted form; encrypted password generated e.g. as $perl\ -e\ 'printf\ crypt("olacomoestas",\ "XA")\ .\ "\backslash n"'$

reboot; do not ask for confirmation for rebooting after installation is completed

Package selection

The list of packages to be installed is inittiated with the %*packages* command. Packages can be specified by component or by individual package name. Components are installed by giving their group name; individual packages may be installed by giving the package name, i.e. their `rpm` file name, excluding the version and platform information.

The available packages are listed in $i386/RedHat/RPMS/$ directory of the installation source. For example, the file

```
expect-5.32.2-67.i386.rpm
```

contains the rpm installation for the package named `expect`, characterized by version (5.32.2; `expect -v`), release (67), architecture (i386).

A list of installed packages is produced during installation, located at $/root/install.log$. One may take as example the package section of the anaconda automatically generated configuration file. This can be modified, but care should be payed to relative dependencies.

pre & *post* install

The pre and post installation sections (started with the %*pre* and %*post* interpreters) are placed at the end of the configuration file, and define shell commands to be run just before and just after the system's installation.

Pre-install commands are run in the installation's image environment.

Post-install commands are run in the change root, system's environment; commands can still be specified to run outside of the chroot environment when the option $--nochroot$ is used. The system's image is accessible on $/mnt/sysimage/$ from the kickstart image (nochroot) environment. Different scripting languages can be activated with the $--interpreter$ option. What is more, one can take advantage of all utilities which have been installed on the newly built Linux system.

Installation and settings of Level3 specific features can be instructed directly using **post** (or **pre**) installation scripting. An alternative, is to use the post install scripting features of workgroups.

nochroot **environment**

The installer image's system is available in the *nochroot* environment. The latter is accessible in pre, $post --nochroot$ (and $after.rpms.nochroot.sh$, when using workgroups) interpreters.

The source $i386$ installation directory is mounted on

```
/mnt/source
```

and the installing system on

```
/mnt/sysimage
```

The workgroup materials, when these are used, are copied to

```
/mnt/sysimage/etc/$WORKGROUP/
```

where `WORKGROUP='cat /etc/workgroup'`, e.g. `CDFlevel3`.

1.1.3 Post configuration

This is the section of the configuration file initiated by the $\%post$ directive, run in chroot (system's) environment.

Here one specifies extra actions to be performed after the normal system installation. In particular, one can access the network, and mount an apppropriate directory.

```
mkdir /mnt/ksdir
mount 192.168.24.11:/home/recovery/kickstart /mnt/ksdir
echo "Kickstart installation performed on ‘/bin/date‘" >  /mnt/ksdir/ks.post
...
umount /mnt/ksdir
```

The list of actions to be performed may include the following.

Update fstab:

```
echo "b0l3serv:/home     /home                    nfs    \
         auto,rw,rsize=8192,wsize=8192,soft" >> /etc/fstab
```

Modify/create general system's configuration files:

```
/etc/hosts
/etc/ntp.conf
/etc/ssh/ssh_config
/etc/ntp/step-tickers
/etc/sysconfig/static-routes
/etc/sysconfig/network
/etc/sysconfig/network-scripts/ifcfg-eth#
```

Perform runtime services settings and others:

```
/sbin/chkconfig --level 35 anacron  on
/sbin/chkconfig --level 35 gpm      on
/sbin/chkconfig --level 35 network  on
/sbin/chkconfig --level 35 sshd     on
/sbin/chkconfig --level 35 netfs    on
/sbin/chkconfig --level 35 atd      on
/sbin/chkconfig --level 35 keytable on
/sbin/chkconfig --level 35 portmap  on
/sbin/chkconfig --level 35 syslog   on
/sbin/chkconfig --level 35 nfslock  on
/sbin/chkconfig --level 35 ypbind   on
/sbin/chkconfig --level 35 crond    on
/sbin/chkconfig --level 35 kudzu    on
/sbin/chkconfig --level 35 random   on
/sbin/chkconfig --level 35 sendmail off
/sbin/chkconfig --level 35 apmd     off

/sbin/hwclock --utc --systohc
/usr/bin/updatedb
```

1.1.4 CDFlevel3 workgroup

Fermi Workgroups [6] have been designed to provide extra installation customization capabilities. In practice, this further extends (and facilitates) the *packages* and *post* installation procedure, automatically running a custom shell script at the end of the install, and providing a *storage* area accessible by the shell script. In particular, this circumvents the need to mount an extra nfs location with Level3 specific materials, as these can be placed in designed locations in the main source location for the installation.

The workgroup is named **CDFlevel3**, and is located in the directory (at the source installation tree location)

`i386/Fermi/workgroups/CDFlevel3`

During installation this gets copied with rpeserved tree structure to $/etc/CDFlevel3$ (in general to /etc/‘cat /etc/workgroup‘), as instructed in the file [1]

`i386/Fermi/common/scripts/post.sh`

with preserved tree structure, thus becoming available to the customization script. It contains the following base directories.

RPMS

Contains workgroup specific rpms to be last installed (with the rpm option $--force$), before customization script is executed.

configfiles

This is a storage area; it may hold farm specific configuration files.

scripts

This is the location for customization scripts. Scripts located in this area are executed in case their names match the following, and have execution permissions:

- before.rpms.sh
- after.rpms.sh
- after.rpms.nochroot.sh

The first two are executed in a chroot (system's) environment; the last one is run in the installing system's context, the install medium location being available in $SOURCE, and the new install area in $CHROOT. In these bash scripts full paths should be specified.

comps

This file defines the groups and rpm packages to be installed during the RedHat packages installation time.

It should be merged with other `comps` files (from other workgroups) and the RedHat defined portion, in the following file.

[1] This script normally is executed automatically. In versions 731x, there was a bug in the installer which prevented this to occur during kickstart, thus not allowing the use of workgroups; the fix, as was first suggested to me by T. Dawson, is to include an explicit call to the script in the post section of the kickstart configuration file:

```
%post –nochroot
sh /mnt/source/Fermi/common/scripts/post.sh
```

```
i386/RedHat/base/comps
```

One may have the following simplest comps definition of the workgroup

```
0 --hide  CDFlevel3 {
  CDFlevel3-tag
  expect
  upsupdbootstrap
}
```

The UPS/UPD installation is completed by placing the $upsupdbootstrap - local * .rpm$ (for installation in $/local$) from $RedHat/RPMS/$ in the workgroup's $RPMS/$ directory.

Farm specific configuration files, as well as necessary tarballs, are placed in $configfiles/$, to be handled by a customization file named $after.rpms.sh$ located in $scripts/$; this would substitute the *post* installation section of the $ks.cfg$ kickstart configuration file.

For RedHat release 8.0 or higher, the comps file, actually **comps.xml**, is written in XML format, for increased flexibility.

The workgroups feature is not *necessary* for Level3 installation, as alternative ways have been designed; nevertheless, it has been set up, fully tested, and is made available, with this section serving as guide, as it may be usefull for additional and administration convenience.

1.2 Installation disk

The kickstart installation may be performed by a floppy disk. This must contain a Linux kernel image and appropriate configuration files; specifically the following files are required:

vmlinuz
initrd.img
ldlinux.sys
syslinux.cfg
ks.cfg

Files `syslinux.cfg` and `ks.cfg` need to be created or customized. The latter has been described in detail in previous sections.

The floppy uses the `syslinux` boot loader; `syslinux.cfg` is its configuration file, and contains the following:

default ks : assign kickstart as default boot image
prompt 0 : do not bring up prompt at boot, or
timeout 60 : decrease delay for default image to be booted
label ks : image label (arbitrary, as long as consistent with default)

kernel vmlinuz : specify name of file containing the kernel
append : add kernel parameters, including
`ks=floppy` : find `ks.cfg` on the floppy (drive `/dev/fd0`)
`initrd=initrd.img` : name of initial root disk image file
`lang=`
`text` : keyword text enables text-based install
`ksdevice=eth0` use this network device to connect to the network
`devfs=nomount ramdisk_size=8192`

The mentioned, customized files should be transferred to a floppy disk. The installation is performed simply by booting the node with the so produced kickstart installation floppy.

During the installation process various consoles are accessible with information of what is taking place:

`Alt-F1` - installation dialog
`Alt-F2` - shell prompt
`Alt-F3` - install log (messages from install program)
`Alt-F4` - system log (messages from kernel, etc.)
`Alt-F5` - other messages

further

In another implementation, for additional functionality, the default image directives in `syslinux.cfg` may be left included; these can be specified at boot prompt, in which case this should not be disabled, and an appropriate delay (e.g., `timeout 60` for 6 seconds) should be specified.

A custom boot message screen may be displayed, by adding the keyword `display ksl3.msg` to the `syslinux.cfg` , where `ksl3.msg` is a message file. The contents of this file may be marked up, such as adding colored words (e.g. `^O09LEVEL3^O02` is displayed blue).

If the line `F1 ksl3.msg` is added, the message is shown when `F1` is pressed at boot time; this allows for displaying several messages, containing for example instructions associated with different kernels or configurations available.

The floppy disk kickstart procedure is most relevant for installing nodes which do not boot up properly, as the installation does not make use of a possibly pre-existing system.

1.3 Hard drive based installation

For nodes which have a running operating system, as is the case during upgrade, a floppy-less, hard drive-based installation procedure may be employed. This is most

convenient, as a node or a pre-defined list of nodes can be installed by execution of a single script, in principle from one's office!

The strategy here is to use the hard drive has the kickstart installation media; it involves the following steps: (i) transfer the necessary images and corresponding configuration files to the hard drive to be installed; (ii) configure the current node's boot loader to use the kickstart image; (iii) reboot the node. After this, if all went well, the nodes come up with a brand new installation.

Currently, LILO is used in the farm; accordingly, its configuration file, `lilo.conf`, is modified by adding the kickstart image specifications.

Assume the node to be installed has the following disk structure: the root filesystem is *hda*1; it has an extra partition, *hda*6 mounted on */usr*; the current directory contains the kickstart image installation files, together with a proper kickstart configuration file (described in previous sections).

```
mkdir /usr/boot
ipadr=‘/sbin/ifconfig eth0 | grep inet        \
       | cut -f 12 -d " " | sed -e s/addr:/""/‘
hostn=‘hostname -s‘
todo=‘echo $ipadr --hostname $hostn‘
sed -e s/"192.168.23.250"/"$todo"/ ks.cfg-template > /usr/boot/ks.cfg
cp -f initrd.img /usr/boot/initrd-install.img
cp -f vmlinuz    /usr/boot/vmlinuz-install
cat <<EOF >>/etc/lilo.conf
  image=/usr/boot/vmlinuz-install
     label=install
     initrd=/usr/boot/initrd-install.img
     append="ks=hd:hda6/boot/ks.cfg"
     read-only
     root=/dev/hda1
EOF
/sbin/lilo
/sbin/lilo -R install
/sbin/reboot
```

1.4 Installation through recovery

Some kinds of node on the farm require specific installation care. this is the case for the Converter and Output nodes. These require extra or modified drivers for additional hardware (e.g., ATM card in Converters, multi-port Ethernet tulip card on both). Two possibilities are foreseen for installing these: (*i*) produce appropriate RPMs, or (*ii*) install the filesystem contents from a properly installed similar node. Given that these are considerably fewer than Processor nodes, and the possibly delicate task of creating proper rpm installation files, the most straightforward way is the latter.

A filesystem recovery based installation can be implemented making use as well of the linux kernel which comes with the Fermi installation. Namely, booting the Linux kernel in its rescue mode

```
boot: linux rescue
```

one is re-directed to a shell environment, allowing to perform general rescue operations. These may include disk inspection, partitioning (e.g. `sfdisk` is available), filesystem creation (e.g. `ext3` feature are available), mouting local and remote filesystems, and so on. Additionally, the `i386` source directory on the installation server is automatically mounted.

The following procedure was therefore implemented and tested.

source directory

Add to the installation source on the server the directory

```
i386/LEVEL3/
```

this becomes accessible in the recovery system's image environment under

```
/mnt/source/LEVEL3/
```

filesystem contents

Place there all source filesystem contents, e.g. in

```
i386/LEVEL3/tarballs/
```

install script

Create a recovery-like script, placing it in the `i386/LEVEL3/` recovery-based installation directory.

```
i386/LEVEL3/scripts/
```

This may include actions as the following:

- `mke2fs -j /dev/$fsname1` : create `ext3` filesystems
- `mkswap -j /dev/$fsswap` : set up swap areas
- `mkdir /mnt/$fsname1; mount -t ext3 /dev/$fsname1 /mnt/$fsname1` : mount filesystems
- `/mnt/$fsname1; tar xvf $tardir/$fstar1`

- `chroot /mnt/$rootfs /sbin/lilo` : configure boot loader
- `echo "Node installed on ‘date‘ > /mnt/$fsname1/l3install.log "` : write installation record

Once such a procedure is set up this way, the installation may be executed by performing the following steps: (*i*) (re-)partition the target disk, if necessary; (*ii*) reboot the node using the installation image in rescue mode (this can be done using a floppy disk, or an hard drive based installation, in a similar fashion as described in previous sections); (*iii*) execute the installation script previously set up,

```
/mnt/source/LEVEL3/scripts/nodeinstall.sh
```

Exiting the rescue shell will reboot the node just installed.

Note that the filesystem table should include explicit partitions locations rather than *label*'s which without further do won't be defined.

References

[1] N. Leonardo for the EVB/L3 team, *Event Builder and Level 3 Manual for Experts*, CDF Note 6138, http://www-cdfonline.fnal.gov/evbl3shift/evbl3pager.html.

[2] N. Leonardo for the EVB/L3 team, *EVB/L3 experts online page*, http://www-cdfonline.fnal.gov/evbl3shift/evbl3pager.html
Recovery, http://www-cdfonline.fnal.gov/evbl3shift/pager/recovery/
Node statistics, http://www-cdfonline.fnal.gov/evbl3shift/pager/recovery/nodelist.html

[3] T. Oehser, *The most GNU/Linux on 1 floppy disk*, http://www.toms.net/rb/

[4] The Femi Linux page, http://www-oss.fnal.gov/projects/fermilinux/
v7.3.1 distribution, http://www-oss.fnal.gov/projects/fermilinux/731/home.html
v7.3.1 ftp, ftp://linux.fnal.gov/linux/731a/
v7.3.1 images, ftp://linux.fnal.gov/linux/731a/i386/images/

[5] The Official Red Hat Linux Customization Guide,
http://www.redhat.com/docs/manuals/linux/RHL-7.3-Manual/custom-guide/index.html

[6] Fermi Workgroups,
http://www-oss.fnal.gov/projects/fermilinux/common/workgroups.maintainers.html

[7] UPS, UPD and UPP page at Fermilab,
http://www.fnal.gov/docs/products/ups/

UPD and UPD distribution server,
http://kits.fnal.gov/

Main ftp server containing available product releases,
ftp://ftp.fnal.gov/products/

A Install configuration and implementation

A.1 Configuration files

File: ks.cfg
Description: main kickstart configuration file

```
nfs --server 192.168.24.11 --dir /scratch/731a.install/i386
install
lang en_US
langsupport en_US
keyboard us
mouse none
text
skipx
network --bootproto static --device eth0 --ip 192.168.23.250                  \
        --netmask 255.255.0.0 --gateway 192.168.24.0 --nameserver 192.168.24.11
xconfig --card "ATI Mach64" --videoram 4096 --hsync 31.5 --vsync 50-61 \
        --resolution 1024x768 --depth 16 --defaultdesktop gnome
rootpw --iscrypted XA8wIytED41RI;
firewall --disabled
authconfig --enableshadow --enablemd5 --enablenis     \
        --nisdomain b0l3p --nisserver b0l3serv.fnal.gov
timezone --utc America/Chicago
bootloader
zerombr yes
clearpart --all --initlabel
part /boot --fstype ext3 --size=50 --ondisk=hda --asprimary
part / --fstype ext3 --size=4096 --ondisk=hda --asprimary
part /tmp --fstype ext3 --size=2048 --ondisk=hda
part swap --size=2047 --ondisk=hda --asprimary
part /cdf --fstype ext3 --size=1 --grow --ondisk=hda
reboot

%packages
@ GNOME
@ Network Support
@ Authoring and Publishing
@ Software Development
@ Kernel Development
@ Fermi Kerberos
@ Openssh Server
@ FermiStandAlone
ghostscript-fonts
balsa
mozilla-chat
compat-libstdc++
gaim
Xaw3d-devel
glade
```

```
libesmtp
ddd
xmms-gnome
libesmtp-devel
libgtop-devel
SDL
gdk-pixbuf-devel
pan
ORBit-devel
doxygen
glib2-devel
kernel-smp
lesstif-devel
libghttp-devel
pygtk-devel
tetex-xdvi
smpeg
gnome-core-devel
bonobo-devel
libcap-devel
gnome-vfs-devel
rsync
SDL_net
control-center-devel
pango-devel
openssh-askpass-gnome
GConf-devel
bonobo-conf-devel
transfig
w3c-libwww-devel
eel-devel
libglade2-devel
xfig
audiofile-devel
xpdf
licq
imlib-devel
gnome-libs-devel
unzip
usbview
gq
gv
gtk+-devel
librsvg-devel
gcc-objc
gal-devel
python2-devel
SDL_image
acroread-plugin
magicdev
w3c-libwww
libpcap
```

```
exmh
fam-devel
freetype-devel
ghostscript
zz_libg2c.a_change
memprof
XFree86-devel
Guppi-devel
lesstif
libglade-devel
xawtv
openssh-askpass
redhat-config-network
ical
guile-devel
libole2-devel
licq-gnome
oaf-devel
nedit
gnome-media
SDL_mixer
librep-devel
gnome-print-devel
atk-devel
libmng-devel
libungif-devel
libxml-devel
acroread
glib-devel
gtk2-devel
netpbm-devel
galeon
ucd-snmp
xmms
emacs
expect


%post

mkdir /mnt/ksdir
mount 192.168.24.11:/home/recovery/kickstart /mnt/ksdir

hostn=`cat /etc/sysconfig/network | grep HOSTNAME | sed -e s/HOSTNAME=//`
touch '/mnt/ksdir/$hostn.kslock'
nodelog='/mnt/ksdir/$hostn.ks.log'
touch $nodelog
echo "Kickstart hd-installation $hostn performed on `/bin/date`"   \
      >>  /mnt/ksdir/ks.log
echo "Starting l3 ks hd install of $hostn on `/bin/date`" >> $nodelog

echo "`/bin/date` : copying config files" >> $nodelog
```

```
echo "b0l3serv:/home     /home                    nfs    \
       auto,rw,rsize=8192,wsize=8192,soft" >> /etc/fstab
cp -f /mnt/ksdir/etc/ks-hosts         /etc/hosts
cp -f /mnt/ksdir/etc/ks-static-routes /etc/sysconfig/static-routes
cp -f /mnt/ksdir/etc/ks-ntp.conf      /etc/ntp.conf
cp -f /mnt/ksdir/etc/ks-ssh_config    /etc/ssh/ssh_config
cp -f /mnt/ksdir/etc/ks-sshd_config   /etc/ssh/sshd_config
cp -f /mnt/ksdir/etc/ks-step-tickers  /etc/ntp/step-tickers

echo "`/bin/date` : copying cdf.tar" >> $nodelog
if [ -d /cdf ]; then
cd /cdf
tar xfvz /mnt/ksdir/cdf/cdf_ks.tgz
cd /
ln -s /cdf/log log
cd /cdf/level3/filter/control
tar xfvz /mnt/ksdir/cdf/ks-control-vtest.tgz
cd /cdf/level3/filter/relay
tar xfvz /mnt/ksdir/cdf/ks-relay-v1_4_7.tgz
fi

echo "`/bin/date` : copying local.tar" >> $nodelog
cd /
tar xpvfz /mnt/ksdir/cdf/ks-local.tgz
#rpm -i --force /mnt/ksdir/rpms/upsupdbootstrap-2.2-8.i386.rpm
#rpm -i --force /mnt/ksdir/rpms/upsupdbootstrap-local-2.2-2.i386.rpm

echo "`/bin/date` : copying ups databases" >> $nodelog
if [ -d /local ]; then
cd /usr/local/etc
ln -s /local/ups/etc/setups.sh .
ln -s /local/ups/etc/setups.csh .
fi

#runtime services settings
/sbin/chkconfig --level 35 anacron  on
/sbin/chkconfig --level 35 gpm      on
/sbin/chkconfig --level 35 network  on
/sbin/chkconfig --level 35 sshd     on
/sbin/chkconfig --level 35 netfs    on
/sbin/chkconfig --level 35 atd      on
/sbin/chkconfig --level 35 keytable on
/sbin/chkconfig --level 35 portmap  on
/sbin/chkconfig --level 35 syslog   on
/sbin/chkconfig --level 35 nfslock  on
/sbin/chkconfig --level 35 ypbind   on
/sbin/chkconfig --level 35 crond    on
/sbin/chkconfig --level 35 kudzu    on
/sbin/chkconfig --level 35 random   on
/sbin/chkconfig --level 35 sendmail off
/sbin/chkconfig --level 35 apmd     off
/sbin/hwclock --utc --systohc
```

```
/usr/bin/updatedb

echo "‘/bin/date‘ : done" >> $nodelog
echo "       Ending installation  of ‘hostname -s‘ on ‘/bin/date‘" \
     >> /mnt/ksdir/ks.log
echo "...." >> /mnt/ksdir/ks.log

touch /root/l3install.log
cat <<EOF >> /root/l3install.log
 Level3 Farm Installation Procedure
 Author: Nuno Leonardo

 Node:   $hostn
 Method: hard-drive kickstart
 Date:   ‘/bin/date‘

 Details:
EOF

cat $nodelog >> /root/l3install.log
rm '/mnt/ksdir/$hostn.kslock'
umount /mnt/ksdir
exit
```

File: syslinux.cfg
Description: floppy boot loader configuration

```
default ks
prompt  1
timeout 10
display ksl3.msg
F1 ksl3.msg
F2 boot.msg
F3 general.msg
F4 rescue.msg
label ks
  kernel vmlinuz
  append ks=floppy ksdevice=eth0 initrd=initrd.img lang= text \
         devfs=nomount ramdisk_size=8192
label linux
  kernel vmlinuz
  append initrd=initrd.img lang= devfs=nomount ramdisk_size=8192 vga=788
label text
  kernel vmlinuz
  append initrd=initrd.img lang= text devfs=nomount ramdisk_size=8192
label expert
  kernel vmlinuz
  append expert initrd=initrd.img lang= devfs=nomount ramdisk_size=8192
label nofb
```

```
  kernel vmlinuz
  append initrd=initrd.img lang= devfs=nomount nofb ramdisk_size=8192
label lowres
  kernel vmlinuz
  append initrd=initrd.img lang= lowres devfs=nomount ramdisk_size=8192
```

File: ksl3.msg

Description: marked up message to be displayed at boot time (instructions may be added).

```
^L

            ^O09Welcome to the ^O0cLevel3 Farm^O09 kickstart installation!^O07


               LL     EEEEEE  V     V EEEEEE LL      333333
               LL     EE      V     V EE     LL           3
               LL     EEEE     V   V  EEEE   LL        3333
               LL     EE        V V   EE     LL           3
               LLLLLL EEEEEE     V    EEEEEE LLLLLL  333333


                    ^O05K   I   C   K   S   T   A   R   T ^O07


                              Procedure by
                            ^O0fNuno T. Leonardo^O07
                         Massachusetts Institute of Technology


            ^O05[F1-Level3] [F2-Main] [F3-General] [F4-Rescue]^O07
```

A.2 Installation scripts

File: ks.harddrive.install.sh

Description: sets node for hard drive based kickstart installation

```
#!/bin/bash
hostn=‘hostname -s‘;
if   [ $hostn = "b0l3pcom1" ] ||  [ $hostn = "b0l3pcom2" ]; then
 echo "ATTENTION: action forbidden in $hostn... goodbye !";
 exit 1;
else
 echo "Node $hostn will be prepared for kickstart HD installation !"
fi
```

```
ipadr=‘/sbin/ifconfig eth0 | grep inet | cut -f 12 -d " " | sed -e s/addr:/""/‘
todo=‘echo $ipadr --hostname $hostn‘
rm -f /boot/ks.cfg
sed -e s/"192.168.23.250"/"$todo"/ /home/leonardo/l3kickstart/ks-hd.cfg > /boot/ks.cfg

cp -f /home/leonardo/l3kickstart/initrd.img /boot/initrd-install.img
cp -f /home/leonardo/l3kickstart/vmlinuz    /boot/vmlinuz-install

if [ -e /etc/lilo.conf-OLD ]; then
echo "lilo.conf will not be modified";
elif [ -e /etc/lilo.conf ]; then
cp -f /etc/lilo.conf /etc/lilo.conf-OLD;
cat <<EOF >>/etc/lilo.conf

#kickstart install image
image=/boot/vmlinuz-install
        label=install
        initrd=/boot/initrd-install.img
        append="ks=hd:hda1/ks.cfg"
        read-only
        root=/dev/hda2
EOF
else
echo "/etc/lilo.conf does not exist \!"
exit;
fi

/sbin/lilo
/sbin/lilo -R install
/sbin/shutdown -r now
exit
```

File: makefloppy.ks.sh
Description: produces node-customized kickstart installation floppy

```
#!/bin/sh
echo "Customized level3 kickstart floppy creation"
echo -n "Enter level3 node name [b0l3ks0]: "
read hostn
echo -n "Complete node IP address 192.168. [23.250]: "
read ipaddr

echo "Customizing configuration..."
if [ ! $hostn ]; then
 hostn='b0l3ks0';
fi
if [ ! $ipaddr ]; then
 ipaddr='192.168.23.250';
```

```
fi
ipaddr="192.168.$ipaddr";

echo "  Node: $hostn   IP: $ipaddr"
sed -e s/"192.168.23.250"/$ipaddr/ ks.cfg-template > ks.cfg-template1
sed -e s/b0l3ks0/$hostn/ ks.cfg-template1 > ks.cfg

echo -n "Insert floppy disk...."
read something

echo -n "Do you want to format floppy? "
read fmat
if [ "$fmat" = "yes" ]; then
echo "Start formatting disk...";
fdformat /dev/fd0H1440
elif [ "$fmat" = "no" ]; then
echo "Skipping disk formatting...";
else
echo "Please try again.";
exit 1;
fi

echo -n "Need to transfer raw image? "
read rimg
if  [ "$rimg" = "yes" ]; then
echo "Transferring image to floppy disk..."
dd if=bootnet.img of=/dev/fd0H1440
elif [ "$rimg" = "no" ]; then
echo "Skipping disk formatting...";
else
echo "Please try again.";
exit 2;
fi

echo "Starting image configuration..."
echo "Mounting disk..."
mount -t msdos /dev/fd0 /mnt/floppy
echo "Costumizing image..."
cp ks.cfg /mnt/floppy
cp syslinux.cfg /mnt/floppy
umount /mnt/floppy
echo "Done"
exit
```

Typical use and output of this script is presented next.

```
> ./makefloppy.sh

Customized level3 kickstart floppy creation
Enter level3 node name [b0l3ks0]: b0l3ks3
Complete node IP address 192.168. [23.250]: 23.253
```

```
Customizing configuration...
  Node: b0l3ks3   IP: 192.68.23.253
Insert floppy disk....
Do you want to format floppy? yes
Start formatting disk...
Double-sided, 80 tracks, 18 sec/track. Total capacity 1440 kB.
Formatting ... done
Verifying ... done
Need to transfer raw image? yes
Transferring image to floppy disk...
2880+0 records in
2880+0 records out
Starting image configuration...
Mounting disk...
Costumizing image...
Done
```

File: nodeinstall.sh

Description: script for node installation through global filesystems recovery using rescue installation image

```
#!/bin/sh

#tarballs location
tardir='/mnt/source/LEVEL3/tarballs'

##### BEGIN of script configuration
#Enter node information:
#node name (change default b0l300):
name=b0l300
#ip address (change default 192.168.23.00)
ipadr=192.168.23.00

#Enter filesystem information; tarballs located on \
# /scratch/731a.install/i386/LEVEL3/tarballs/
#filesystem 1: root
fsname1=hda2
fstar1=rootfs_246.731.tar
fstar1a=usr.731.tar
fsip1=192.168.23.46
#filesystem 2: boot
fsname2=hda1
fstar2=boot.731.tar
#filesystem 3: cdf
fsname3=hda6
fstar3=cdf.731.tar
fstar3a=ks-control-vtest.tar
fstar3b=ks-relay-v1_4_7.tar
#filesystem: swap
```

```
fsswap=hda3
#filesystem: tmp
fstmp=hda5

echo "The node to be installed is $name with ip address $ipadr"

echo "Recreating all filesystems ..."
mke2fs -j /dev/$fsname1
mke2fs -j /dev/$fsname2
mke2fs -j /dev/$fsname3
mke2fs -j /dev/$fstmp
mkswap -j /dev/$fsswap

echo "Mount all partitions..."
mkdir /mnt/$fsname1
mkdir /mnt/$fsname2
mkdir /mnt/$fsname3
mount -t ext3 /dev/$fsname1 /mnt/$fsname1
mount -t ext3 /dev/$fsname2 /mnt/$fsname2
mount -t ext3 /dev/$fsname3 /mnt/$fsname3

echo "Untarring ..."

cd /
echo "Untar directories: root"
cd /mnt/$fsname1
tar xvf $tardir/$fstar1
if [ ! -d /mnt/$fsname1/usr ]; then
mkdir /mnt/$fsname1/usr
fi
if [ ! -d /mnt/$fsname1/boot ]; then
mkdir /mnt/$fsname1/boot
fi
if [ ! -d /mnt/$fsname1/tmp ]; then
mkdir /mnt/$fsname1/tmp
fi
if [ ! -d /mnt/$fsname1/home ]; then
mkdir /mnt/$fsname1/home
fi

cd /mnt/$fsname1/usr
tar xvf $tardir/$fstar1a

cd /
echo "Untar directories: boot"
cd /mnt/$fsname2
tar xvf $tardir/$fstar2

cd /
echo "Untar directories: cdf"
cd /mnt/$fsname3
tar xvf $tardir/$fstar3
```

```
cd /mnt/$fsname3/level3/filter/control
tar xvf $tardir/$fstar3a
cd /mnt/$fsname3/level3/filter/relay
tar xvf $tardir/$fstar3b

cd /
echo "Changing IP address in ifcfg-eth0 ..."
cd /mnt/$fsname1/etc/sysconfig/network-scripts
sed -e s/$fsip1/$ipadr/ ifcfg-eth0 > tempo
mv tempo ifcfg-eth0

echo "Installed node $name; recovered $fsname1 $fsname2 $fsname3; \
    on `/usr/bin/date` " > /mnt/\$fsname1/root/l3install.log

echo "Update mbr"
umount /mnt/$fsname2
mount -t ext3  /dev/$fsname2 /mnt/$fsname1/boot
chroot  /mnt/$fsname1 /sbin/lilo

echo "Un-mounting filesystems..."
umount /mnt/$fsname1/boot
umount /mnt/$fsname2
umount /mnt/$fsname3

echo "Probing floppy disk..."
mkdir /mnt/floppy
mount -t msdos /dev/fd0 /mnt/floppy
flp=`df | grep floppy | wc | cut -f 1`
umount /mnt/floppy
while [ $flp -gt 0  ]; do
 sleep 5
mount -t msdos /dev/fd0 /mnt/floppy
 flp=`df | grep floppy | wc | cut -f 1`
 echo "Still waiting for floppy disk to be removed... $flp : not zero"
 umount /mnt/floppy
done
echo "Floppy drive is now empty."
echo "Done."
echo "The node is now being rebooted..."
sleep 2
reboot
```